IRACST – International Journal of Computer Networks and Wireless Communications (IJCNWC), ISSN: 2250-3501
Vol.12, No 3, July– Sep 2022

Cloud Computing Initiative using Modified AntColony Framework

Soumya Banerjee, Indrajit Mukherjee, and P.K. Mahanti

***Abstract*—**Scheduling of diversified service requests in distributed computing is a critical design issue. Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtual computers. It is not only the clusters and grid but also it comprises of next generation data centers. The paper proposes an initial heuristic algorithm to apply modified ant colony optimization approach for the diversified service allocation and scheduling mechanism in cloud paradigm. The proposed optimization method is aimed to minimize the scheduling throughput to service allthe diversified requests according to the different resource allocator available under cloud computing environment.

***Keywords*—**Ant Colony, Cloud Computing, Grid, Resource allocator, Service Request.

INTRODUCTION

THE initial application of grid computing was functional after considering the advancement of several IT enabled distributed utility services like water, electricity, gas, and telephony for customers. Grid's pervasiveness, ease of use, and reliability [1][8], motivated the use of Grid computing and it was initially driven by large-scale, resource (computational and data)-intensive scientific applications. Those applications demand more resources than a single computer (PC, workstation, supercomputer, or cluster). Today, the latest paradigm of grid to emerge is that of Cloud computing [2] which promises reliable services delivered through next- generation data centers. These components are built on compute and storage virtualization technologies. Consumers will be able to access applications and data from a "Cloud" anywhere in the world on demand. In other words, the Cloud appears to be a single point of access for all the computing needs of consumers. The consumers are assured that the Cloud infrastructure is very robust and will always be available at any time. Envisaging the architecture of the cloud, the present paper proposes an idea to manage cloud computing using ant colony optimization (ACO). The relevance of ant colony to cloud computing architecture is emphasized in the paper, as ant colony also works well in distributed environment like grid. By definition, "cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through multi- negotiation between the service provider and consumers". Hence, there is substantial scope to employ multi-objective optimization strategies among the cloud oriented or grid-based services. The multi-objective optimization could be initiated through ant colony, which is comparatively cost effective approach and simple to implement. To measure the throughput of heterogeneous computing system the term *Makespan* has been introduced. The objective of the ant colony based cloud computing initiative is to minimize the makespan. It is well known that the problem of deciding on an optimal assignment of requests to resources allocator is (Non Polynomial) NP- hard. We develop a heuristic algorithm based on modified ant colony optimization to solve this problem.

The rest of the paper is organized as follows: Section II describes brief background and correlation of cloud computing and ant colony. Section III elaborates the proposed algorithm and steps to implement it on the cloud architecture followed by the result and implications in section IV. Finally, section V gives conclusion and mentions further scope of research in thisdirection.

BACKGROUND OF CLOD COMPUTING AND RELEVANCE WITH ANT COLONY

Cloud computing is a term used to describe both a platform and type of application. A cloud computing platform dynamically provisions, configures, reconfigures, and de provisions servers as needed. Servers in the cloud can be physical machines or virtual machines. Advanced clouds typically include other computing resources such as storage area networks (SANs), network equipment, firewall and other security devices. Cloud computing also describes applications that are extended to be accessible through the Internet. These *cloud applications* use large data centers and powerful servers that host Web applications and Web services. Anyone with a suitable Internet connection and a standard browser can access a cloud application. The primary components of cloud architecture are:

Users/Brokers: Users or brokers acting on their behalf submit service requests from anywhere in the world to the Data Center and Cloud to be processed.

SLA (Service Level Agreements) Resource Allocator: The SLA Resource Allocator acts as the interface between the Data Center/Cloud service provider and external users/brokers. It requires the interaction of the defined scheduled mechanisms to support SLA-oriented resource management.

Google App Engine [3] allows a user to run Web applications written using the Python programming language. Other than supporting the Python standard library, Google App Engine also supports Application Programming Interfaces (APIs) for the data store, Google Accounts, URL fetch, image manipulation, and email services. Google App Engine also provides a Web-based Administration Console for the user to easily manage his running Web applications. Currently, Google App Engine is free to use with up to 500MB of storage and about 5 million page views per month.

Microsoft Live Mesh [4] aims to provide a centralized

location for a user to store applications and data that can be accessed across required devices (such as computers andmobile phones) from anywhere in the world. The user is able to access the uploaded applications and data through a Web- based Live Desktop or his own devices with Live Meshsoftware installed. Each user's Live Mesh is password- protected and authenticated via his Windows Live Login, while all file transfers are protected using Secure Socket Layers (SSL).

Both the applications of Google and Microsoft cloud initiative can be divided into divided into three phases [5]. There are phases involved in resource recovery, scheduling, and executing. In the second phase find the best match between the set of jobs and available resources. The second phase is a NP-hard Problem [6]. The computational grid is a dynamic and unpredictable behavior. They are:

- Computational performance of each resource variesfrom time to time.
- The connection between computers and mobile phones may be unreliable.
- The resources may join or relinquish the grid at anytime
- The resource may be unavailable without a notification.

The scheduling of cloud architecture is dynamic in nature andmoreover Grid middleware and applications are using local scheduling and data co-scheduling. The approach of replication has been also applied and assisted in scheduling and optimization of replication. There are different existing algorithms like the Genetic algorithm (GA) is used for searching large solution space. On other hand, simulated Annealing (SA) is an iterative technique that considers only one possible solution for each meta-task at a time.

ACO algorithm can be interpreted as parallel replicated Monte Carlo (MC) systems [7]. MC systems are general stochastic simulation systems, that is, techniques performing repeated sampling experiments on the model of the system under consideration by making use of a stochastic component in the state sampling and/or transition rules. Experimental results are used to update some statistical knowledge about theproblem. In turn, this knowledge can be also iteratively used to reduce the variance in the estimation of the described variables and directing the simulation process toward the most interesting state space regions. Analogously, in ACO algorithms the ants sample the problem's solution space by repeatedly applying a stochastic decision policy until a

feasible solution of the considered problem is found. The sampling is realized concurrently by a collection of differently instantiated replicas of the same ant type. Each ant "experiment" allows to adaptively modifying the local statistical knowledge on the problem structure. The algorithmis recursive in nature.

PROPOSED ALGORITHM

The classic ant colony algorithm can be described as follows [7]:

Step 1. Initialize
Step 2. Loop /* An iteration */Step 3. Each ant is positioned on a starting node.
Loop /* A step */
Step 4. Each ant applies a state transition rule to incrementally build a solution and a local pheromone updating rule until all ants have built a complete solution Step 5. Global pheromone updating rule is applied until end condition.
Step 6. Stop further iterations

Each edge between node (r, s) has a distance or cost associate δ (r, s) and a pheromone concentration τ (r, s). The equation 1 is the state transition rule, which is a probabilistic function for each node u, which has not been visited by each placed ant on node r.

$$P_k(r,s) = \frac{[\tau(r,s)][\eta(r,s)]^\beta}{\Sigma[\tau(r,u)][\eta(r,u)]^\beta} \quad (1)$$

The parameter β determine the relevance of the pheromoneconcentration compared with the distance or cost, τ (r, s) Global pheromone updating rule can be applied as:

$$\tau(r,s) = (1-\alpha)\tau(r,s) + \Sigma\Delta\tau_k(r,s) \quad (2)$$

Where α is the pheromone evaporation factor between 0 and 1 and $\Delta\tau_k$ (r, s) is the reverse of the distance or cost done by ant k, if (r, s) is its path and is 0 if it is not in the path.

The steps can be modified to manage cloud architecture. The cloud is visualized is the collection of clustered services, hence the live services of cloud behaves like an ant, when it find its file object, the ant died. Subsequently, considering theprime component of cloud computing, the *compute cloud* and*storage cloud* can be modeled as virtual services of cloud.

Every time a request is processed on a cloud cluster site, τ is updated for all the site connections and thus the "(2)" can bemodified by associating a parameter $\dot{t}$.

$$\dot{t}(r,s) = (1-\alpha_{tcs})\tau(r,s) + \sum\Delta\tau_{k+tcs}(r,s) \quad (3)$$

The dot operator represents time for each cloud scheduling service. Therefore, the α_{tcs} is introduced which expressesthe evaporation factor under time slot of cloud service. The heuristic can be divided into two categories for cloud-based services e.g. on-line mode service and the batch mode service. In online mode, whenever a request arrive, it immediately allocate to the first free resource allocator. The arrival order of the request in cloud is important in this method. Here, each service request is considered only once for matching and scheduling. In batch mode, the requests are collected; the

scheduler considers the approximate execution time for each task and use heuristic approach to possibly make better decision.

The function free [j] – return time, when the resource allocators M_j will be free. We consider,

$$free\ [j] = I_{\Delta} + ET_{ij,} \quad (4)$$

where, I_{Δ} is the initial time slot of request of service made on request r_i on resource allocator m.

The scheduling of resource allocator on the cloud service proposes the probability of servicing the request:

$$p_{ij} = \frac{ph_{ij}\eta_{ij}(1/ET_{ij})}{\sum ph_{ij} - \eta_{ij}(1/ET_{ij})} \quad (5)$$

Where,

- η_{ij} is the attractiveness of the move as computed by heuristic information indicating a prior desirability of that move.
- ph_{ij} is the pheromone trail level of the move, indicating the fast and accuracy of the cloud service in the past (with lower α_{tcs}) to make that particular move (it represents therefore a posterior service accomplishment indication of the desirability of that request)
- ET_{ij} - Execution Matrix of service and resource allocator.

In this proposed model, we select the highest probability's 'i' and 'j' are the next request of service r_i executed on the resource allocator j.

I. Result and Implementation

We have developed simulated cloud examples (formulated in python code version 2.5.2. under P-IV machine) with derived from Google App Engine and Microsoft Live Mesh to evaluate the proposed modified ACO algorithm for scheduling request and its respective services on cloud architecture. Practically, in this model, we incorporate 5 diversified resource allocators like Application Programming Interfaces (APIs) for the data store, Google Accounts, URL fetch, image manipulation, and email services. Google App Engine also provides a Web-based Administration Console for the user with 25 different service requests on these utilities. The initial parameters are set as follows: $\tau_0 = 0.01$ and $\rho = 0.5$ and we useunit ant per services.

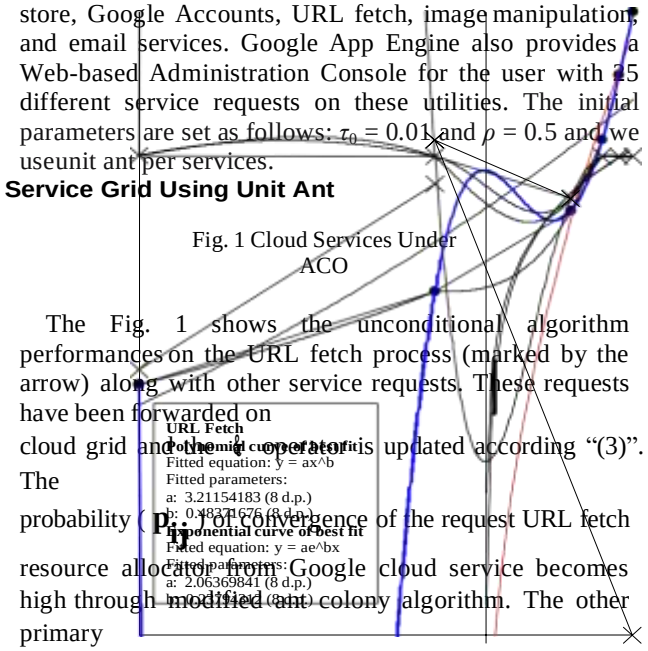

Fig. 1 Cloud Services Under ACO

The Fig. 1 shows the unconditional algorithm performances on the URL fetch process (marked by the arrow) along with other service requests. These requests have been forwarded on cloud grid and the p_{ij} operator is updated according "(3)". The probability (p_{ij}) of convergence of the request URL fetch resource allocator from Google cloud service becomes high through modified ant colony algorithm. The other primary resource allocators are Google Accounts, image manipulation, and email services and Application Programming Interfaces (APIs) for the data store. The proposed ACO improve performances on the half of the simulation series, considering all these resource allocators are active at the same time. Other services are plotted in Fig. 1 in different characteristics curves like logarithmic curve and exponentiation curve for other resource allocator's services shown in Table I.

TABLE I
MAKESPAN FOR THE EXECUTION ON FIRST FREE MACHINE AND ACO ALGORITHM

Resource Allocator		Classic ACO	Modified ACO
URL Fetch	240	199	140
Image Manipulation	101	73	70
E mail Service	241	171	114
Google Acc.	141	101	91
Google API	133	99	77

The Table I compare result achieved with existing ACO algorithm for similar service requests [5] and result derived from the proposed modified ACO. The results are

measured in minutes. The Service response Time is from 0.5 units to 200 units and service grid is the equal to the value of factorial of 5 defined resource allocators (shown in Table I).

It is shown that only the modified ¢ and time slot distribution under different cloud services provides a better performance. For conventional online-mode the arriving order of request is very important, whereas for both modified and classis ant colony most important is the execution time of the separate request and free time of resource allocator to service the request

CONCLUSION

In this paper, a heuristic algorithm based on modified ant colony optimization has been proposed to initiate the service load distribution under cloud computing architecture. The pheromone update mechanism of ACO and coefficient τ is modified to ¢. This modification supports to minimize the makespan of the cloud computing based services and probability of servicing the request also has been converged using the modified scheduling (Refer Equation "5"). The simulation doesn't consider the fault tolerance issues. Due to absence of any restore time in service and resource allocator distribution, it is expected that continuous ant colony with other modified parameters could demonstrate better results compared to other optimization models, even in faulty service request and disrupted resource allocator.

REFERENCES

Chu, K. Nadiminti, C. Jin, S. Venugopal, and R. Buyya. Aneka: "Next- Generation Enterprise Grid Platform for e-Science and e-Business Applications", in *Proceedings of the 3th IEEE International Conference on e-Science and Grid Computing (e-Science 2007)*, Bangalore, India, Dec. 2007.

A. Weiss, "Computing in the Clouds", *netWorker*, Volume 11, No.4, pp.16-25, December, 2007.

Google App Engine, http://appengine.google.com [accessed in March 2009]

Microsoft Live Mesh, http://www.mesh.com [accessed in March 2009]

Stefka Fidanova and Mariya Durchova," Ant Algorithm for Grid Scheduling Problem", Large Scale Computing, Lecture Notes in Computer Science No. 3743, Springer, , pp 405-412, 2006.

Yaohang Li, "A bio-inspired adaptive Job Scheduling Mechnism on a Computational Grid", International Journal of Computer Science and Network Security, Vol.6.No.3.B, March 2006.

Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: "Optimization by a colony of cooperating agents", *IEEE Transactions on Systems", Man, and Cybernetics*, part B, 26(1) pp. 1–13, 1996.

S. Venugopal, X. Chu, and R. Buyya, "A Negotiation Mechanism for Advance Resource Reservation using the Alternate Offers Protocol." in *Proceedings of the 16th International Workshop on Quality of Service (IWQoS 2008)*, Twente, The Netherlands, June 2008.